Assignment 2

Student Name:

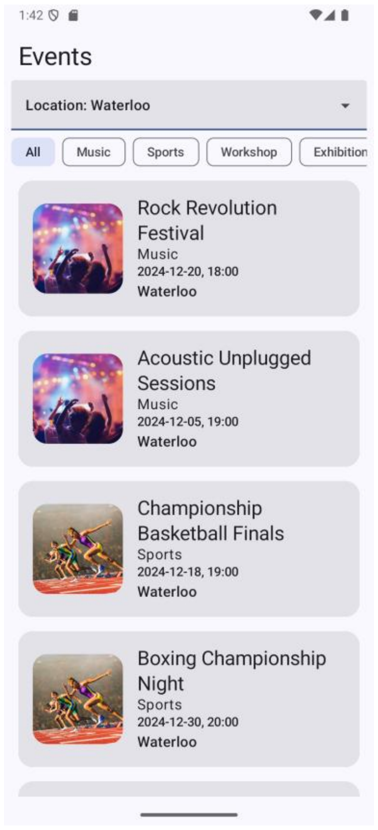
Parth Gajjar (8959999)

Course:

PROG3320 - Systems Project Planning & Initiation

Faculty: Thomas Methew

Screenshots (Light and Dark Color Scheme):



1:42

Events

Location: Waterloo

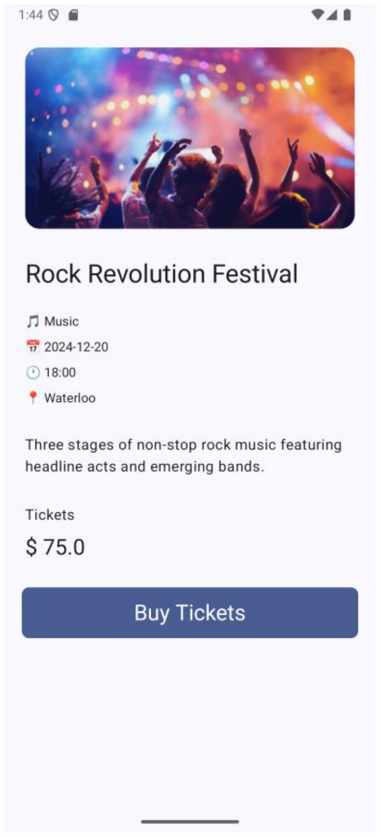
All | Music | Sports | Workshop | Exhibition

Rock Revolution Festival
Music
2024-12-20, 18:00
Waterloo

Acoustic Unplugged Sessions
Music
2024-12-05, 19:00
Waterloo

Championship Basketball Finals
Sports
2024-12-18, 19:00
Waterloo

Boxing Championship Night
Sports
2024-12-30, 20:00
Waterloo



1:44

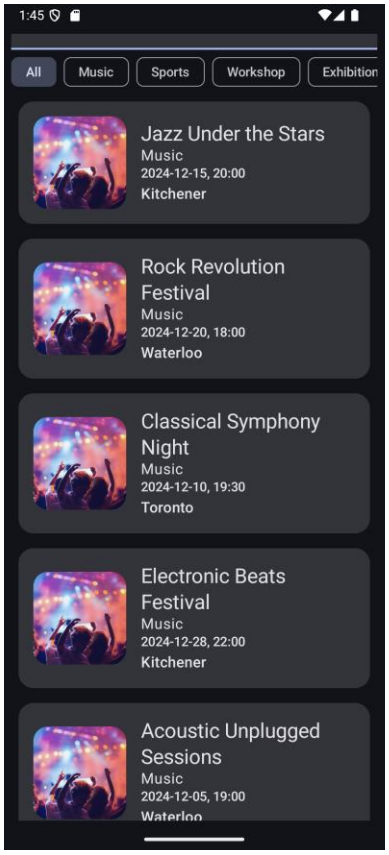
Rock Revolution Festival

Music
2024-12-20
18:00
Waterloo

Three stages of non-stop rock music featuring headline acts and emerging bands.

Tickets
$ 75.0

Buy Tickets



1:45

All | Music | Sports | Workshop | Exhibition

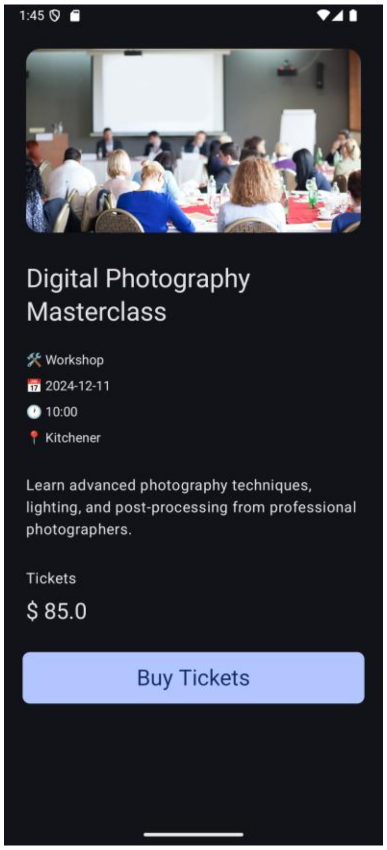
Jazz Under the Stars
Music
2024-12-15, 20:00
Kitchener

Rock Revolution Festival
Music
2024-12-20, 18:00
Waterloo

Classical Symphony Night
Music
2024-12-10, 19:30
Toronto

Electronic Beats Festival
Music
2024-12-28, 22:00
Kitchener

Acoustic Unplugged Sessions
Music
2024-12-05, 19:00
Waterloo



1:45

Digital Photography Masterclass

Workshop
2024-12-11
10:00
Kitchener

Learn advanced photography techniques, lighting, and post-processing from professional photographers.

Tickets
$ 85.0

Buy Tickets

Code:

Event.kt:

```
package com.example.eventexplorer8959999.data

// Event Model
data class Event (
   val name: String,
   val imageRes: Int,
   val date: String,
   val time: String,
   val location: String,
   val description: String,
   val ticketPrice: Float,
   val category: EventCategories
)
```

Event Categories:

```
package com.example.eventexplorer8959999.data

// Enum for Event Categories
enum class EventCategories {
   Music,
   Sports,
   Workshop,
   Exhibition
}
```

SampleEvents.kt:

```
package com.example.eventexplorer8959999.data

import com.example.eventexplorer8959999.R

// Sample data for events
val sampleEvents = listOf(
   // Music Events
   Event(
      name = "Jazz Under the Stars",
      imageRes = R.drawable.music_event,
      date = "2024-12-15",
      time = "20:00",
      location = "Kitchener",
      description = "An enchanting evening of smooth jazz featuring local and international artists under
the open sky.",
      ticketPrice = 45.00f,
      category = EventCategories.Music
   ),
   Event(
      name = "Rock Revolution Festival",
      imageRes = R.drawable.music_event,
      date = "2024-12-20",
      time = "18:00",
      location = "Waterloo",
      description = "Three stages of non-stop rock music featuring headline acts and emerging bands.",
      ticketPrice = 75.00f,
      category = EventCategories.Music
   ),
   Event(
      name = "Classical Symphony Night",
      imageRes = R.drawable.music_event,
      date = "2024-12-10",
      time = "19:30",
      location = "Toronto",
      description = "Experience Beethoven's greatest works performed by the City Philharmonic
Orchestra.",
      ticketPrice = 60.00f,
      category = EventCategories.Music
   ),
   Event(
      name = "Electronic Beats Festival",
      imageRes = R.drawable.music_event,
      date = "2024-12-28",
      time = "22:00",
      location = "Kitchener",
      description = "Dance all night to the best DJs in electronic music with stunning visual effects.",
      ticketPrice = 55.00f,
      category = EventCategories.Music
   ),
   Event(
      name = "Acoustic Unplugged Sessions",
      imageRes = R.drawable.music_event,
      date = "2024-12-05",
      time = "19:00",
      location = "Waterloo",
      description = "Intimate acoustic performances in a warm, candlelit atmosphere with local singer-
songwriters.",
      ticketPrice = 25.00f,
      category = EventCategories.Music
   ),
   Event(
      name = "Hip Hop Block Party",
      imageRes = R.drawable.music_event,
      date = "2024-12-22",
      time = "17:00",
      location = "Toronto",
      description = "Street culture celebration featuring live rap battles, breakdancing, and graffiti art.",
      ticketPrice = 30.00f,
      category = EventCategories.Music
   ),

   // Sports Events
   Event(
      name = "City Derby Football Match",
      imageRes = R.drawable.sport_event,
      date = "2024-12-12",
      time = "15:00",
      location = "Kitchener",
      description = "The biggest rivalry match of the season between the city's top two football clubs.",
      ticketPrice = 50.00f,
      category = EventCategories.Sports
   ),
   Event(
      name = "Championship Basketball Finals",
      imageRes = R.drawable.sport_event,
      date = "2024-12-18",
      time = "19:00",
      location = "Waterloo",
      description = "Watch the thrilling conclusion to this season's basketball championship series.",
      ticketPrice = 65.00f,
      category = EventCategories.Sports
   ),
   Event(
      name = "Marathon City Run",
      imageRes = R.drawable.sport_event,
      date = "2024-12-08",
      time = "07:00",
      location = "Toronto",
      description = "Join thousands of runners in the annual city marathon with 5K, 10K, and full marathon
options.",
      ticketPrice = 35.00f,
      category = EventCategories.Sports
  ),
  Event(
      name = "Tennis Open Tournament",
      imageRes = R.drawable.sport_event,
      date = "2024-12-14",
      time = "11:00",
      location = "Kitchener",
      description = "World-class tennis action featuring top-ranked players competing for the
championship title.",
      ticketPrice = 80.00f,
      category = EventCategories.Sports
  ),
  Event(
      name = "Boxing Championship Night",
      imageRes = R.drawable.sport_event,
      date = "2024-12-30",
      time = "20:00",
      location = "Waterloo",
      description = "Witness epic heavyweight championship bouts with rising stars and veteran fighters.",
      ticketPrice = 95.00f,
      category = EventCategories.Sports
  ),
  Event(
      name = "Swimming Gala Finals",
      imageRes = R.drawable.sport_event,
      date = "2024-12-06",
      time = "18:00",
      location = "Toronto",
      description = "Olympic-style swimming competition showcasing the fastest swimmers in the region.",
      ticketPrice = 40.00f,
      category = EventCategories.Sports
  ),

  // Workshop Events
  Event(
      name = "Digital Photography Masterclass",
      imageRes = R.drawable.workshop_event,
      date = "2024-12-11",
      time = "10:00",
      location = "Kitchener",
      description = "Learn advanced photography techniques, lighting, and post-processing from
professional photographers.",
      ticketPrice = 85.00f,
      category = EventCategories.Workshop
  ),
  Event(
      name = "Pottery and Ceramics Workshop",
      imageRes = R.drawable.workshop_event,
      date = "2024-12-16",
      time = "14:00",
      location = "Waterloo",
      description = "Hands-on pottery workshop where you'll create your own ceramic pieces from
scratch.",
      ticketPrice = 70.00f,
      category = EventCategories.Workshop
  ),
  Event(
      name = "Culinary Arts: French Cuisine",
      imageRes = R.drawable.workshop_event,
      date = "2024-12-19",
      time = "17:00",
      location = "Toronto",
      description = "Master the art of French cooking with a Michelin-starred chef teaching classic
techniques.",
      ticketPrice = 120.00f,
      category = EventCategories.Workshop
  ),
  Event(
      name = "Web Development Bootcamp",
      imageRes = R.drawable.workshop_event,
      date = "2024-12-09",
      time = "09:00",
      location = "Kitchener",
      description = "Intensive day-long workshop covering HTML, CSS, and JavaScript fundamentals for
beginners.",
      ticketPrice = 95.00f,
      category = EventCategories.Workshop
  ),
  Event(
      name = "Watercolor Painting Class",
      imageRes = R.drawable.workshop_event,
      date = "2024-12-23",
      time = "13:00",
      location = "Waterloo",
      description = "Explore watercolor techniques and create beautiful landscape paintings in a
supportive environment.",
      ticketPrice = 55.00f,
      category = EventCategories.Workshop
  ),
  Event(
      name = "Sustainable Gardening Workshop",
      imageRes = R.drawable.workshop_event,
      date = "2024-12-27",
      time = "10:00",
      location = "Toronto",
      description = "Learn organic gardening methods, composting, and how to grow your own vegetables
sustainably.",
      ticketPrice = 45.00f,
      category = EventCategories.Workshop
  ),

  // Exhibition Events
  Event(
      name = "Modern Art Retrospective",
      imageRes = R.drawable.exhibition_event,
      date = "2024-12-07",
      time = "10:00",
      location = "Kitchener",
      description = "A comprehensive exhibition showcasing 100 years of modern art movements and
masterpieces.",
      ticketPrice = 20.00f,
      category = EventCategories.Exhibition
  ),
  Event(
      name = "Ancient Civilizations Gallery",
      imageRes = R.drawable.exhibition_event,
      date = "2024-12-13",
      time = "09:00",
      location = "Waterloo",
      description = "Discover artifacts and treasures from ancient Egypt, Greece, and Rome in this
fascinating exhibition.",
      ticketPrice = 25.00f,
      category = EventCategories.Exhibition
  ),
  Event(
      name = "Contemporary Photography Showcase",
      imageRes = R.drawable.exhibition_event,
      date = "2024-12-17",
      time = "11:00",
      location = "Toronto",
      description = "Stunning photographic works from award-winning contemporary photographers
around the world.",
      ticketPrice = 15.00f,
      category = EventCategories.Exhibition
  ),
  Event(
      name = "Science and Innovation Expo",
      imageRes = R.drawable.exhibition_event,
      date = "2024-12-21",
      time = "10:00",
      location = "Kitchener",
      description = "Interactive exhibition featuring cutting-edge technology, robotics, and scientific
discoveries.",
      ticketPrice = 30.00f,
      category = EventCategories.Exhibition
  ),
  Event(
      name = "Wildlife Photography Awards",
      imageRes = R.drawable.exhibition_event,
      date = "2024-12-26",
      time = "10:00",
      location = "Waterloo",
      description = "Breathtaking images of wildlife from around the globe, celebrating nature's beauty and
diversity.",
      ticketPrice = 18.00f,
      category = EventCategories.Exhibition
  ),
  Event(
      name = "Sculpture Garden Opening",
      imageRes = R.drawable.exhibition_event,
      date = "2024-12-29",
      time = "12:00",
      location = "Toronto",
      description = "New outdoor exhibition featuring contemporary sculptures by renowned artists set in
beautiful gardens.",
      ticketPrice = 22.00f,
      category = EventCategories.Exhibition
  )
)
```

EventCard.kt:

```
package com.example.eventexplorer8959999.components

import androidx.compose.foundation.Image
import androidx.compose.foundation.clickable
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.layout.width
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material3.Card
import androidx.compose.material3.CardDefaults
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import com.example.eventexplorer8959999.R
import com.example.eventexplorer8959999.data.Event
import com.example.eventexplorer8959999.data.EventCategories
import com.example.eventexplorer8959999.ui.theme.EventExplorer8959999Theme
```

```
@Composable
fun EventCard(event: Event, modifier: Modifier = Modifier, onEventClick: (Event) -> Unit) {

  // Event Card Component
  Card (
    modifier = modifier
      .fillMaxWidth()
      .padding(8.dp)
      .clip(RoundedCornerShape(16.dp))
      .clickable{
        onEventClick(event)
      },
    elevation = CardDefaults.cardElevation(6.dp),
    shape = MaterialTheme.shapes.large
  ) {
    Row(
      modifier = modifier.padding(16.dp),
      verticalAlignment = Alignment.CenterVertically
    ) {
      // Event Image
      Image(
        painter = painterResource(id = event.imageRes),
        contentDescription = event.name,
        contentScale = ContentScale.Crop,
        modifier = modifier
          .size(100.dp)
          .clip(RoundedCornerShape(16.dp))
      )

      Spacer(modifier = Modifier.width(16.dp))

      // Event Info: Name, Category, Date, Location
      Column (
        modifier = Modifier.weight(1f)
      ) {
        Text(text = event.name, style = MaterialTheme.typography.titleLarge)
        Text(text = event.category.toString(), style = MaterialTheme.typography.bodyLarge)
        Text(text = "${event.date}, ${event.time}", style = MaterialTheme.typography.titleSmall)
        Text(text = event.location, style = MaterialTheme.typography.titleMedium)

      }
    }
  }
}

@Composable
@Preview(showBackground = true)
fun EventCardPreview(){
  val sampleEvent = Event(
    name = "Jazz Under the Stars",
    imageRes = R.drawable.music_event,
    date = "2024-12-15",
    time = "20:00",
    location = "Blue Note Theater",
    description = "An enchanting evening of smooth jazz featuring local and international artists under
the open sky.",
    ticketPrice = 45.00f,
    category = EventCategories.Music
  )

  EventExplorer8959999Theme {
    EventCard(event = sampleEvent, onEventClick = {})
  }
}
```

EventDetailScreen.kt:

```
package com.example.eventexplorer8959999.features.eventdetail

import androidx.compose.foundation.Image
import androidx.compose.foundation.gestures.scrollable
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.layout.width
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.DateRange
import androidx.compose.material3.Button
import androidx.compose.material3.Icon
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.lifecycle.viewmodel.compose.viewModel
import com.example.eventexplorer8959999.R
import com.example.eventexplorer8959999.features.eventlist.EventListViewModel

@Composable
fun EventDetailScreen(viewModel: EventListViewModel, onBackClick: () -> Unit, modifier: Modifier =
Modifier){

  val event = viewModel.selectedEvent
  var scrollState = rememberScrollState()

  // Event Detail page
  Surface (
    modifier = modifier
      .fillMaxSize()
      .padding(8.dp),
  ) {
    Column (
      modifier = modifier
        .fillMaxWidth()
        .padding(8.dp)
        .verticalScroll(scrollState),
      horizontalAlignment = Alignment.CenterHorizontally,
      verticalArrangement = Arrangement.spacedBy(16.dp)
    ) {
```

```kotlin
// Event Logo
Image(
  painter = painterResource(event.imageRes),
  contentDescription = event.name,
  contentScale = ContentScale.Crop,
  modifier = modifier
    .fillMaxWidth()
    .padding(8.dp)
    .size(200.dp)
    .clip(RoundedCornerShape(16.dp))
)

// Event Details
Column (
  modifier = modifier
    .fillMaxWidth()
    .padding(8.dp),
  verticalArrangement = Arrangement.spacedBy(8.dp)
) {
  // Title
  Text(event.name, style = MaterialTheme.typography.headlineMedium)

  // Category, Date, Time, Location
  Spacer(modifier.height(8.dp))
  Text(
    "${viewModel.selectedEventCategoryIcon} " + event.category,
    style = MaterialTheme.typography.bodyMedium
  )
  Text("📅 " + event.date, style = MaterialTheme.typography.bodyMedium)
  Text("🕐 " + event.time, style = MaterialTheme.typography.bodyMedium)
  Text("📍 " + event.location, style = MaterialTheme.typography.bodyMedium)

  // Description
  Spacer(modifier.height(16.dp))
  Text(event.description, style = MaterialTheme.typography.bodyLarge)


  // Ticket Price
  Spacer(modifier.height(16.dp))
  Text(stringResource(R.string.event_ticket_price_title))
  Text("$ ${event.ticketPrice}", style = MaterialTheme.typography.headlineSmall)
}
```

```
      // Buy Tickets Button
      Button(
        onClick = {},
        modifier = modifier
          .fillMaxWidth()
          .padding(4.dp)
          .height(56.dp),
        shape = RoundedCornerShape(8.dp)
      ) {
        Text(stringResource(R.string.event_buy_button), style =
MaterialTheme.typography.headlineSmall)
      }
    }
  }
}
```

EventListScreen.kt:

```
package com.example.eventexplorer8959999.features.eventlist

import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.width
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.LazyRow
import androidx.compose.foundation.lazy.items
import androidx.compose.material3.Button
import androidx.compose.material3.DropdownMenu
import androidx.compose.material3.DropdownMenuItem
import androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.ExposedDropdownMenuBox
import androidx.compose.material3.ExposedDropdownMenuDefaults
import androidx.compose.material3.FilterChip
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.MenuAnchorType
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.material3.TextField
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.lifecycle.viewmodel.compose.viewModel
import com.example.eventexplorer8959999.R
import com.example.eventexplorer8959999.components.EventCard
import com.example.eventexplorer8959999.data.Event
import com.example.eventexplorer8959999.ui.theme.EventExplorer8959999Theme

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun EventListScreen(viewModel: EventListViewModel, onEventClick: () -> Unit, modifier: Modifier =
Modifier) {
  // Event List Screen
  Surface(
    modifier = Modifier.fillMaxSize(),
    color = MaterialTheme.colorScheme.background
  ) {
    LazyColumn (
      modifier = Modifier.padding(8.dp)
    ) {
      // Page Title
      item {
        Text(
          text = stringResource(R.string.event_list_title),
          style = MaterialTheme.typography.headlineMedium,
          modifier = Modifier.padding(8.dp)
        )
      }

      // Location Filter Dropdown Menu
      item{
        Spacer(Modifier.width(8.dp))
        OptIn(ExperimentalMaterial3Api::class)
        ExposedDropdownMenuBox(
          expanded = viewModel.isLocationMenuExpanded,
          onExpandedChange = {
```

```
viewModel.updateLocationMenu(!viewModel.isLocationMenuExpanded)}
        ) {
          TextField(
            value = "Location: ${viewModel.selectedLocation}",
            onValueChange = {},
            readOnly = true,
            trailingIcon = {
              ExposedDropdownMenuDefaults.TrailingIcon(
                expanded = viewModel.isLocationMenuExpanded
              )
            },
            modifier = Modifier
              .menuAnchor(type = MenuAnchorType.PrimaryEditable, enabled = true)
              .fillMaxWidth(),
            textStyle = MaterialTheme.typography.titleMedium
          )
          ExposedDropdownMenu(
            expanded = viewModel.isLocationMenuExpanded,
            onDismissRequest = { viewModel.updateLocationMenu(false)}
          ) {
            viewModel.locations.forEach { location ->
              DropdownMenuItem(
                text = { Text( text = location) },
                onClick = {
                  viewModel.onLocationSelected(location)
                  viewModel.updateLocationMenu(false)
                }
              )
            }
          }
        }
      }

      // Category Filter Chip Buttons
      item {
        Spacer(Modifier.width(8.dp))

        LazyRow(
          horizontalArrangement = Arrangement.spacedBy(8.dp)
        ) {
          items(viewModel.categories) { category ->
            val isSelected = viewModel.selectedCategories.contains(category)
```

```
                FilterChip(
                    selected = isSelected,
                    onClick = { viewModel.toggleCategory(category) },
                    label = { Text(category) }
                )
            }
        }
    }

    // Event Cards
    items(viewModel.filteredEvents) { event ->
        EventCard(
            event = event,
            onEventClick = {
                viewModel.onEventSelected(event)
                onEventClick()
            }
        )
    }
  }
 }
}

@Composable
@Preview(showBackground = true)
fun EventListScreenPreview() {
    val viewModel : EventListViewModel = viewModel ()
    EventExplorer8959999Theme {
        EventListScreen(viewModel = viewModel, onEventClick = {})
    }
}
```

EventListViewModel:

```
package com.example.eventexplorer8959999.features.eventlist

import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Info
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.setValue
import androidx.lifecycle.ViewModel
import com.example.eventexplorer8959999.data.Event
import com.example.eventexplorer8959999.data.sampleEvents

class EventListViewModel : ViewModel() {
  val events = sampleEvents   // Sample Events

  // Selected Event to pass to event detail screen
  var selectedEvent by mutableStateOf<Event>(events[0])
    private set

  // Switch Event to show in detail screen
  fun onEventSelected(event: Event) {
    selectedEvent = event
    selectedEventCategoryIcon = categoryIcons[event.category.toString()] ?: ""
  }

  // Collect location dynamically based on all unique locations in events list
  val locations = listOf("All") +  events.map { it.location }.distinct()

  // Expanded/Closed location menu state
  var isLocationMenuExpanded by mutableStateOf(false)
    private set
  fun updateLocationMenu(expanded: Boolean) {
    isLocationMenuExpanded = expanded
  }

  // Selected Location to filter events list by location
  var selectedLocation by mutableStateOf("All")
    private set

  fun onLocationSelected(location: String) {
    selectedLocation = location
  }

  // Categories List
  val categories = listOf("All", "Music", "Sports", "Workshop", "Exhibition")

  // Category Icons
  val categoryIcons = mapOf(
    "Music" to "🎵",
    "Sports" to "🏈",
    "Workshop" to "🛠️",
    "Exhibition" to "🏢"
  )

  // Selected Categories to filter events list by category
  var selectedCategories by mutableStateOf(setOf<String>("All"))
    private set

  // Currently selected event's icon for event detail screen
  var selectedEventCategoryIcon by mutableStateOf("")
    private set

  // Filtered Events by selected Location and Categories
  val filteredEvents: List<Event>
    get(){
        // Location filtering
        var filtered = if(selectedLocation == "All") {
            events
        } else {
            events.filter { it.location == selectedLocation }
        }

        // Category filtering
        if (!selectedCategories.contains("All")) {
            filtered = filtered.filter { it.category.toString() in selectedCategories }
        }

        return filtered
    }

  // Toggle Categories for event list
  fun toggleCategory(category: String) {
    val newSelectedCategories = selectedCategories.toMutableSet()
    if (category == "All") {
        newSelectedCategories.clear()
        newSelectedCategories.add("All")
    } else {
        newSelectedCategories.remove("All")
        if (newSelectedCategories.contains(category)) {
            newSelectedCategories.remove(category)
        } else {
            newSelectedCategories.add(category)
        }
```

```
            if (newSelectedCategories.isEmpty() || newSelectedCategories.size == categories.size - 1) {
                newSelectedCategories.clear()
                newSelectedCategories.add("All")
            }

        }
        selectedCategories = newSelectedCategories
    }
}
```

AppNavigation.kt:

```
package com.example.eventexplorer8959999.navigation

import android.annotation.SuppressLint
import androidx.compose.runtime.Composable
import androidx.compose.runtime.remember
import androidx.compose.ui.Modifier
import androidx.lifecycle.viewmodel.compose.viewModel
import androidx.navigation.NavController
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import com.example.eventexplorer8959999.features.eventdetail.EventDetailScreen
import com.example.eventexplorer8959999.features.eventlist.EventListScreen
import com.example.eventexplorer8959999.features.eventlist.EventListViewModel

@SuppressLint("UnrememberedGetBackStackEntry")
@Composable
fun AppNavigation(navController: NavController, modifier: Modifier = Modifier){

    // Routes
    NavHost(
        navController = navController as androidx.navigation.NavHostController,
        startDestination = "event_list", // Default Route
        modifier = modifier
    ) {
        // Event List Screen Route
        composable("event_list"){
            val viewModel: EventListViewModel = viewModel()
            EventListScreen(
                viewModel = viewModel,
                onEventClick = {
```

```
                navController.navigate("event_detail")
            }
        )
    }

    // Event Detail Screen Route
    composable("event_detail") {
        // Get ViewModel from backstack entry of event_list screen
        val eventListStackEntry = remember { navController.getBackStackEntry("event_list") }
        val eventListViewModel: EventListViewModel = viewModel(eventListStackEntry)
        EventDetailScreen(
            viewModel = eventListViewModel,
            onBackClick = { navController.popBackStack() }
        )
    }
  }
}
```

MainActivity.kt:

```
package com.example.eventexplorer8959999

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.tooling.preview.Preview
import androidx.navigation.compose.rememberNavController
import com.example.eventexplorer8959999.navigation.AppNavigation
import com.example.eventexplorer8959999.ui.theme.EventExplorer8959999Theme

class MainActivity : ComponentActivity() {
  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    enableEdgeToEdge()
    setContent {
            EventExplorer8959999Theme {
                val navController = rememberNavController()
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    // App Navigator to handle navigation
                    AppNavigation(navController = navController, modifier = Modifier.padding(innerPadding))
                }
            }
        }
    }
}
```

strings.xml

```
<resources>
    <string name="app_name">EventExplorer8959999</string>
    <string name="event_buy_button">Buy Tickets</string>
    <string name="event_ticket_price_title">Tickets</string>
    <string name="event_list_title">Events</string>
</resources>
```